SampComp: sample-based techniques for algorithmic composition

Andrea Valle
StudiUm/CIRMA - Università di Torino
andrea.valle@unito.it

ABSTRACT

The paper introduces a series of techniques for algorithmic composition in which compositional data are extracted from audio at the sample level. In this way, typical Digital Signal Processing operations can act as tools to create/retrieve music materials. First, in order to describe audio-driven processes in composition, the notion of audioparity is introduced. Secondly, various approaches are discussed that map sample values to composition events. The implementation of these processes is then presented, in particular in relation to music notation generation. Finally, the paper discusses some applications of the same techniques to physical computing.

1. INTRODUCTION: AUDIOPARITY, AUDIOPARISM AND TIME SCALING

Algorithmic approaches and techniques to music (regardless of the final medium, whether directly the audio signal, or the musical notation or messages for multimedia objects) are based on a preliminary, general assumption: the presence of some formally defined procedure able to produce a number of representations of musical data. Since the work of Chomsky on formal grammars [1], this kind of approach is typically labelled "generativism", and in the artistic domain it is so strictly tied to the use of computational resources that historically the terms "computer art" and "generative art" are synonyms [2]. In Chomsky's approach, the term has a technical meaning (generativism as a formal grammatical theory), but its Latin etymology is indeed linked to human/animal reproduction. In this regard, zoology classically distinguishes between viviparity (the offspring is generated alive), oviparity (the embryonic development takes place in the deposited egg), and ovoviviparity (there are eggs, but they hatch in the mother's body, and the offspring is born alive). The distinction is interesting because it concerns the ways in which the reproductive behaviour takes place, and is orthogonal to zoological classes (notoriously, the platypus is at the same time mammal and oviparous). In relation to these aspects, the mode of (re-) production and the independence from the class, one could think of a taxonomy of the modes of musical creation, regardless of styles, supports or eras. Consequently, as the suffix "-parity" is used in zoology to indicate reproduction, "audioparity" may be proposed to indicate a composition mode in which musical data originate from sound. A compositional technique is "audioparous" if it defines a projection between a source sound material and an outgoing musical organization. In short, one has audioparity when a composition c results from the application of an audioparous function F_{ap} on a sound s: thus, $c = F_{ap}(s)$. Instead, "audioparism" could be used to describe an aesthetic and practical composition attitude centred on audioparity. Such a category does not depend on authors or periods, and, in the same historical context, different composition practices may be more or less audioparous. As an example, French Renaissance composer Clément Janequin has often pursued a pervasive audioparism in many pieces (*La chasse, La bataille, Les chants des oyseaulx*), explicitly inspired by the transcription of acoustic events. Similarly, in the 20th Century Olivier Messiaen has notoriously integrated the ornithological transcription into many of his works, but often including it into a different composition context [3]. In contemporary music, Spectralism is evidently an area of choice for a pervasive audioparism (see [4]). In his now classic *Les espaces acoustiques* cycle, Gérard Grisey literally re-orchestrates sounds by taking into account data from sonograms: as the French composer said, "we are musicians and our model is sound" (cit. in [5]). On the other side, many composition practices are non-audioparous at all. As an example, Twelve-tone technique is indeed not audioparous, as it organises in various ways an abstract source material, i.e. a 12 pitch-class structure. But the same argument applies to most polyphonic music from 14th to 16th Century. Without discussing further whether the notion of audioparity is relevant or not from a musicological perspective, an interesting point in many audioparous practices, like the ones introduced before, concerns time scale. Messiaen already noted that bird songs must be slowed down in order to match a comfortable time resolution for human ear [3]. In the same pathway, temporal dilatation in relation to the original sound material source is a crucial feature in Grisey's work. This process of time expansion is not irrelevant, because of non-linearities in time perception. While discussing such an issue in the musical context and following the seminal reflections by Stockhausen [6], Roads notes that, as sound passes from one time scale to another, it crosses perceptual boundaries, as human perception processes each time scale differently [7]. In relation to this, Xenakis has proposed to distinguish four time boundaries (macro/meso/mini/microstructures), that divide time scale into separate perceptual regions, i.e. timbres, notes/events,

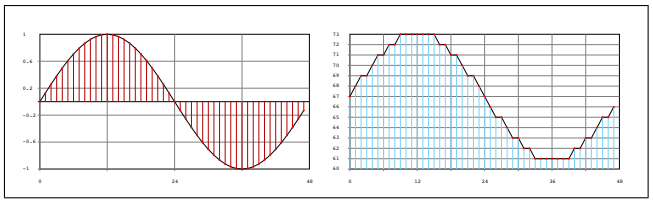

Figure 1. Sinusoidal signal src in a 48-point table (left); requantized version mld (right).

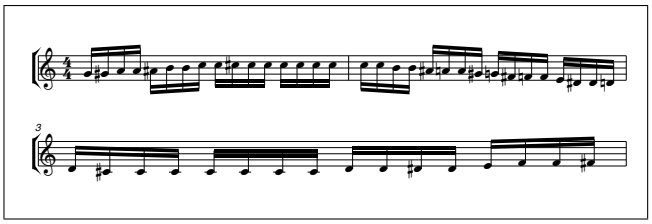

Figure 2. Transcription in music notation of Figure 1 (1 semiquaver per sample).

rhythms, global evolutions [8]. While these considerations are indeed relevant from a perceptual perspective, still, from the point of view of composition practices, they do not prevent to translate a given feature at a certain time level into another one, as it happens in the aforementioned works by Messiaen or Grisey. The following sections will therefore discuss a set of audioparous techniques that translate sonic microstructures (specifically, at the sample level) into sonic ministructures (at the event/note level). These techniques can be called SampComp, as they relate the sample level to the composition one.

2. SAMPCOMP TECHNIQUES

By providing an abstract unified representation for all data, the numerical domain allows to easily switch between time scales in sound representation. SampComp processes, while in some way related to those digital synthesis techniques that focus on the time-domain representation and operate directly at the sample level (e.g. Xenakis, Koenig, Brün, see [9]), still have with a different goal, that of a time scale expansion. The idea that the signal can act as a control of "notational" parameters is widely used in analog electronic music, for example in Voltage Control technique. As an example, Strange's 1972 work [10] has investigated the subject by discussing voltage control operations in relation to standard music notation. However, the numerical domain is more pervasive and flexible and allows to precisely map sample data into note ones. As a general case, the following discussion will focus on the generation of a sequence of pitches (i.e. a melody) from a waveform.

The simplest case may take into account a sine wave, as the most basic and usual signal in audio synthesis. The generated digital signal (a cycle) is represented by a vector containing the values in normalised range ($[\pm 1.0]$), as this is a standard representation, for example widely used for waveform tables in digital oscillators. The resulting digital signal src, for an array of 48 elements (an arbitrary size), is plotted in Figure 1, left.

Two basic digital operations can be applied to the src signal. The first is resampling. In particular, since the amount of information that describes a signal over time is typically very large if compared to the one describing a melody (e.g. 1 second of audio at CD quality requires 44100 samples), the signal can be resampled at a lower rate than the original. The second is quantization. Similar to sampling, the information of the source signal src (in floating point format) has typically a much higher resolution than the one required for pitch description. Quantization therefore defines the range within which the sample values from src are mapped onto pitches. For example, if we consider to map 32 bit audio samples onto an equally-tempered octave (12 pitches), the resolution of the source signal must be reduced to less than 4 bits (as $2^4 = 16 > 12$). These two operations structurally introduce a loss of information and, in fact, a quantity of noise, as they are non-linear. But the exploration of resampling and quantization error is indeed a large part of the interest of the operation. The simplest case consists in resampling by taking a sample every n. The new sampling rate srn will be an integer submultiple of the original sampling sr, i.e. the sampling period sTn will be an integer multiple of sT. As far as quantization is concerned, a mapping function can be defined that associates the value in the sample domain ($[\pm 1.0]$) to values in a certain range in the integer pitch one by linear interpolation. In Figure 1, on the right, the src signal is converted into the new mld signal [1], through a mapping onto the octave that has the middle G as a pivot (hence, in MIDI notation, in the range $[67 \pm 6]$, from 61 to 73) [2], and with $srn = sr$ (all 48 samples of src are taken into account). From this simple example two potentially interesting compositional features emerge:

- to some extent, the sinusoidal waveform of the signal src is maintained as the overall form of the melody in the new signal mld;
- the quantization to the equally tempered octave introduces a distortion that results in a *terracing* operation, so to speak.

The transcription of mld into musical notation (Figure 2) assumes that every sample in mld has a duration of one semiquaver.

Expanding further the discussion, the src signal in Figure 3, left, again counts 48 samples, but this time 3 cycles of the sine wave are tabulated. The mapping quantization function is the same, but the mld signal is obtained by resampling src with a sampling period $sTn = 3 \times sT$ (that is, one sample of src is taken every three). As seen in Figure 3, right, the sinusoidal waveform is still visible, but

[1] Sample values are traced in light blue for sake of visibility.

[2] Each float value in the range $[-1.0, 1.0]$ obtained from src is mapped onto the range $[61, 73]$ by means of $(\frac{(v+1)}{2}) \times 12 + 61$, then rounded and converted to integer. The same mapping will be applied in all following examples for sake of simplicity. Some audio examples are available here:
`http://www.musicaelettronica.it/campcomp-lineamenti-di-musica-audiopara-2/`

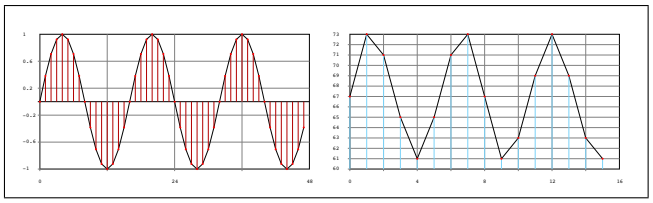

Figure 3. 3 cycles of a sinusoidal signal (left); requantized version (right).

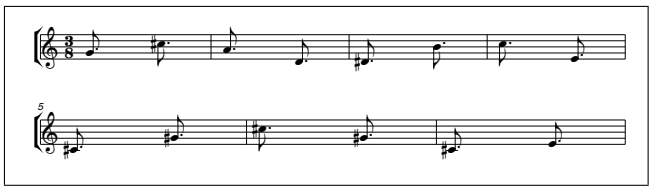

Figure 4. Transcription in music notation of Figure 3 (1 semiquaver per sample).

the combined effect of the presence of 3 cycles and sampling/quantization produces a set of asymmetries in the resulting melody. The musical transcription (Figure 4) may decide to assign any duration to any sample of mld, but to underline the relationship between sT and sTn it still assumes the equivalence between a sample of src and a semiquaver.

Proceeding further, the following example demonstrates the increase in asymmetry by increasing the number of cycles in src (5 in 100 tabulated samples, Figure 5) and resampling with $sTn = 7$ (a sampling that is not yet critical, but certainly important). Although in some way the resulting pitch sequence still bears a sinusoidal shape, the former –as a result of the relationship between resampling and quantization in mld– includes only 9 pitches, 4 of which introduced in a sort of final ending tail, while the other 5 are repeated twice (3 times in the case of the C♯, MIDI note 61). It can be seen that the resulting pattern is made up of 15 elements. As in the previous examples, the transcription into music notation (Figure 6) maintains the equivalence between source samples and semiquavers: the period $sTn = 7$ can easily counted as a duration of 7 semiquavers in each note.

The previous examples have shown that the underlying logic in SampComp is to explore signal time-patterns at the sample level and exploit them at the mini/meso level. This working hypothesis allows then to explore standard Digital Signal Operations (DSP) techniques originally devised for signal generation and manipulation (and well-known in

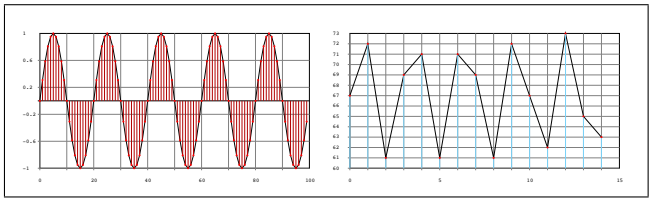

Figure 5. 5 cycles of a sinusoidal signal (left); requantized version (right).

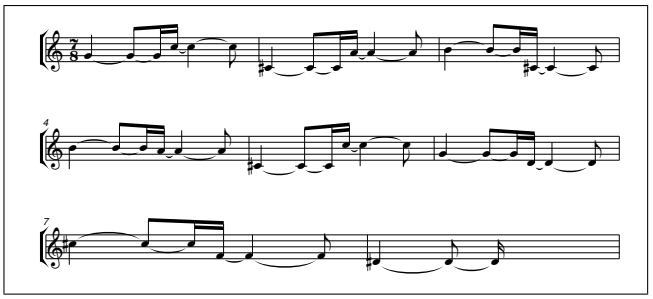

Figure 6. Transcription in music notation of Figure 5 (1 semiquaver per sample).

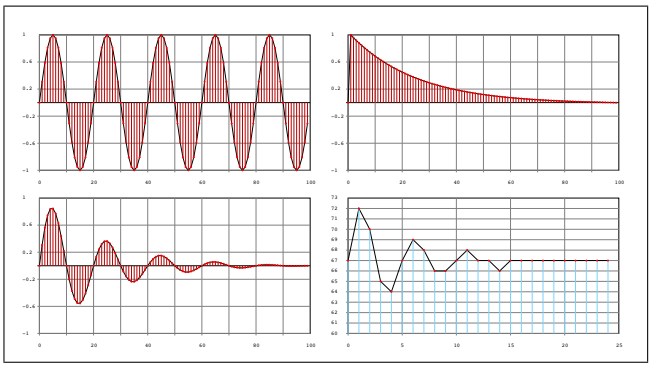

Figure 7. Sinusoidal signal (top, left), percussive envelope (top, right), enveloped sinusoid (bottom, left); requantized version (bottom, right).

electronic music) for the purposes of melodic generation. As such operations are defined in the time domain, a basic case is the application of an amplitude envelope. In Figure 7 the src signal (bottom, left) is obtained by multiplying 5 cycles of a sinusoid in 100 samples (top, right) by a percussive (i.e. exponentially decreasing) envelope signal (top, left). The melodic contour mld (with $sTn = 4 \times sT$, 1 value extracted every 4 samples, Figure 7, bottom, right), that result from mapping, clearly shows the relationship between the signal amplitude of src and the distribution along the chosen range for pitches. The fading amplitude tail of src results in the iteration of the pivot pitch G (MIDI 67). Transcription into musical notation is given in Figure 8.

The application of an envelope can be generalized. The construction of the waveform thus becomes the construction of the melodic contour, in relation to sampling and quantization. In Figure 9, on the left, a 100-point tabulated sinusoid undergoes three operations. The first is a

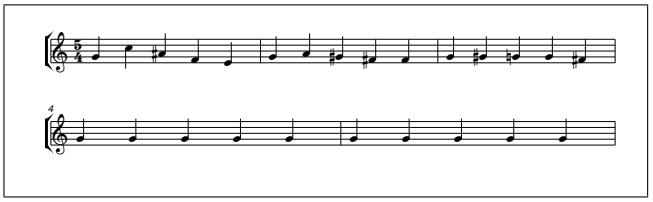

Figure 8. Transcription in music notation of Figure 7 (1 semiquaver per sample).

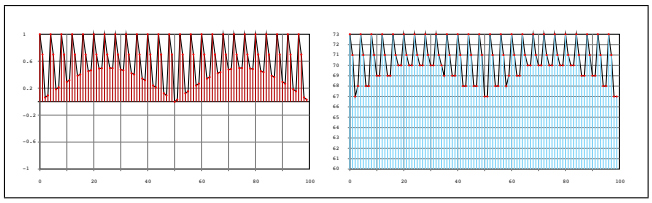

Figure 9. Sinusoidal signal processed by absolute value and impulse replacing (left); requantized version (right).

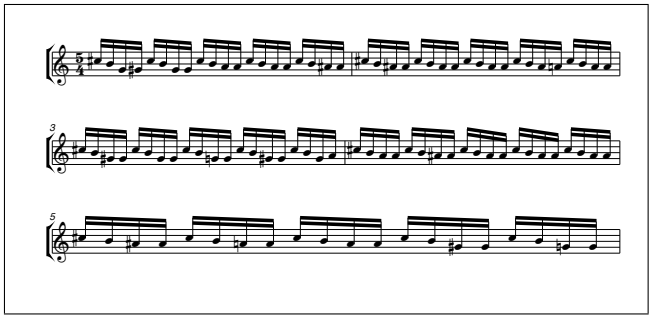

Figure 10. Transcription in music notation of Figure 9 (1 semiquaver per sample).

form of digital distortion through the application of the absolute value that reverses the negative half-cycle to positive values (in the audio domain, this process has been proposed by Puckette [11] as a crude but efficient octaver algorithm that introduces distortion). The second is a rescaling of the amplitude (here halved) so that it is compressed in the range $[0.0, 0.5]$. Finally, a further operation is applied, iterated with modulo 4 on the samples, which replaces the first and second values of a four-sample block with 1 and 0.7 respectively. Such an operation, while typically numerical, can however be thought of as a form of mixing by replacing, in which samples of the sine-derived wave are regularly replaced by an almost impulsive positive signal. Such a process can be easily generalized, i.e. in terms of mixing of weighted sum of signal vectors to obtain desired shapes (or, with an experimental attitude, to explore emerging melodic shapes from various possible operations). Even in the compressed selected pitch range (an octave) the result in terms of melodic transcription (for $sTn = sT$, Figure 10) is interesting as it shows some perceptual features related to grouping. The melody in fact tends to segregate into two streams [12], the pivot top note and the lower "terraced" arpeggio, in a sort of virtual polyphony (the melodic contour is more clearly displayed by Figure 9, right).

In the previous discussion, a fixed resampling rate for src has been assumed. While fixed sampling frequency is a standard technique in DSP, SampComp techniques may benefit from non standard approaches. In this regard, variable sampling can be applied to a src signal. In electronics, a Sample & Hold circuit polls the signal, extracts the value, and keeps it until a new polling is asked for [13]. The next example is based on a Sample & Hold algorithm that polls the source signal src according to a sequence of variable sample intervals instead of a constant rate. Figure 11 is

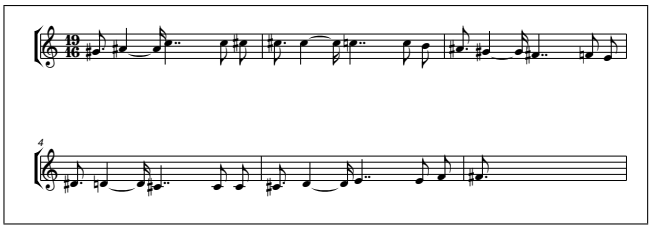

Figure 11. Musical transcription of a variable rate Sample & Hold algorithm applied to one cycle of a sinusoidal signal.

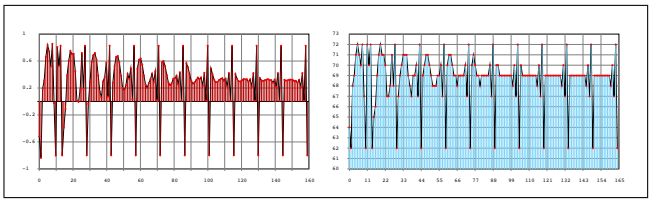

Figure 12. A signal obtained by iteratively chaining averaged 15-sample blocks (left); requantized version (right).

the transcription into music notation that results from such a process. Here, a 100-point 1-cycle sinusoid src signal is sampled along the (cyclic) series $[3, 5, 7, 2, 2]$, i.e. the samples $3, 5, 7, 2, 2$ are taken from src, and so on. By counting note durations in semiquavers in Figure 11, the "rhythmic" sampling becomes apparent.

All the previous examples have started from a sinusoid signal as the most basic signal in electronic music. Indeed, the manipulation of signals in the time domain opens up a Pandora's box of possibilities, far more extensive than the previous preliminary considerations. Further explorations may take into account a large corpus of DSP techniques in order to generate note contour material. In the following, a simple case is thus introduced as an example. An averaging filter is an operation that calculates the value of a sample x through the average of a certain number n of nearby samples (before or after x) [14]. The recursive application of a filter is at the base of the notorious Karplus-Strong algorithm [15], in which a noisy signal is stored into a table that is rewritten iteratively by applying a median filter (the so-called "recirculating wavetable"). In the following ex-

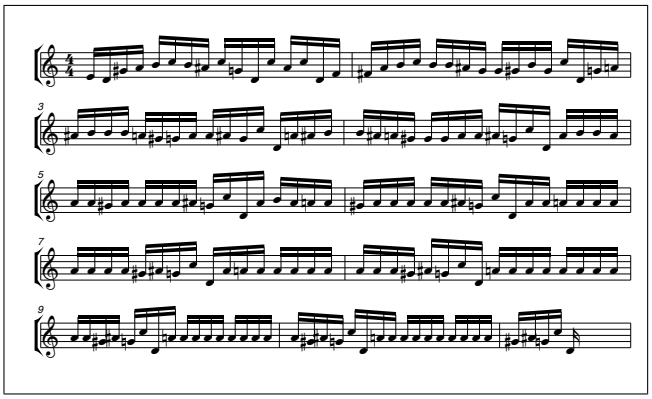

Figure 13. Transcription in music notation of Figure 12 (1 semiquaver per sample).

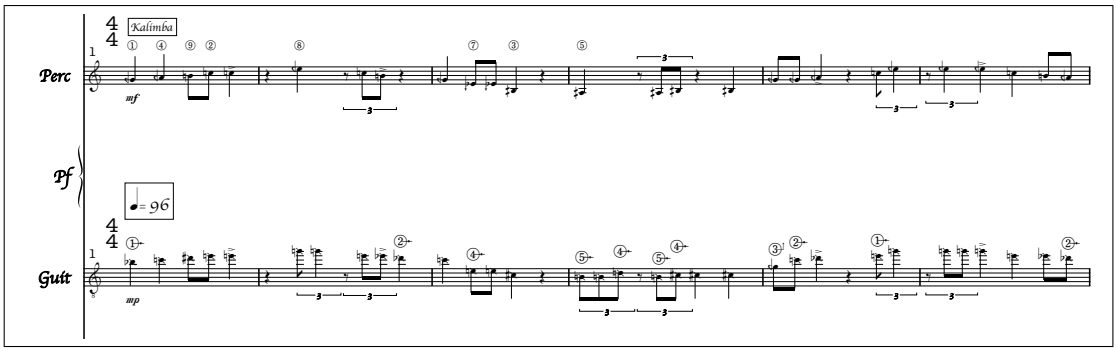

Figure 14. Beginning of *Alatia*, from *Regnum lapideum*.

ample, a src signal is purposely constructed in a two-step process. At the initialisation step, a signal src_0 of 15 points is filled with random values. Then, in the second step, the signal src_0 is filtered (in fact, it is a low pass filter) with a step of 3 samples: the value of x_n is calculated by averaging the samples $[x_n, ..., x_{n+2}]$. The new 15-sample src_1, that results from filtering, is concatenated with the previous one, then src_1 is filtered again by the same procedure to obtain src_2, and so on. The filtering and chaining process is repeated 10 times, so that the final src signal has a size of 165 samples. As shown in Figure 12, the recursive filtering progressively smoothes the edges of the curve: in the mapped melodic signal mld, the initial dispersion of the pitches is thus progressively replaced by a more or less stable curve, centred around a pivot note, as it can be seen in Figure 13. The final result depends crucially on the randomly filled sample block src_0 at the initialization step.

In order to conclude the discussion with a real musical example, Figure 14 shows the beginning of the piece *Alatia*, from the work *Regnum lapideum* by the author and Mauro Lanza, commissioned in 2016 by Ensemble 2e2m (see later). Here the melodic contour is assigned to kalimba and (prepared) guitar. A variable Sample & Hold technique such as the one introduced before has been used to generate the pitch profile. It is easy to recognise a sinusoidal pitch pattern, obtained with a variable ("rhythmic") polling rate.

3. IMPLEMENTATION

The implementation of SampComp techniques by itself does not pose particular problems as the numerical procedures involving the manipulation of small vectors are neither particularly complex nor they require relevant computational resources. All previous examples have been implemented in the SuperCollider environment [16], which features audio synthesis capabilities and an Object-Oriented language to control them. The use of a programming language specifically dedicated to audio is particularly appropriate because the latter comes with a large repertoire of dedicated audio algorithms that can be easily integrated into the workflow. Moreover, through SuperCollider it is possible to sonify the obtained signals and visualise them through graphical interfaces for sake of inspection. A crucial point in algorithmic composition techniques that include musical notation concerns the integration of automatically generated

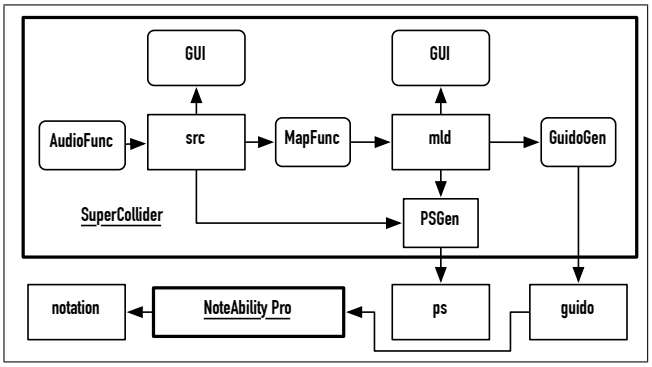

Figure 15. Integrated software pipeline for SampComp and notation generation.

notation in the workflow [17]. In fact, an experimental approach such as SampComp requires that the generation of the notation is integrated into the compositional process in order to rapidly inspect the final results. To this purpose, a light software architecture has been implemented. In Figure 15, the softwares involved are boxed in bold, the software modules are shown in rounded corners, generated data structures and files are enclosed in rectangles. As shown in the Figure 15, SuperCollider implements the generation of the src signal (through specialised modules, in Figure labeled as AudioFunc) and the mapping functions (MapFunc) that result in data structures for the representation of pitches (mld). SuperCollider also operates as a "glueing" solution. The PSGen module generates text files in the PostScript vector graphic format [18] for the displaying of the generated signals (both src and mld). The graphic visualisations of signals shown in the previous Figures have been obtained in this way. The integration of music notation into an automated pipeline is not a trivial task. An option could be to generate MIDI files as an intermediate format to be imported into music notation software, the main drawback being the loss of control on music notation typesetting, as the task is left to the importing software. On the other side, MusicXML [3] requires the definition of a large set of typesetting-related parameters. Another option for the generation of musical notation is LilyPond [19], a music engraving software which features a text-based input. Rather, in the present work the musical examples

[3] https://www.w3.org/2017/12/musicxml31/

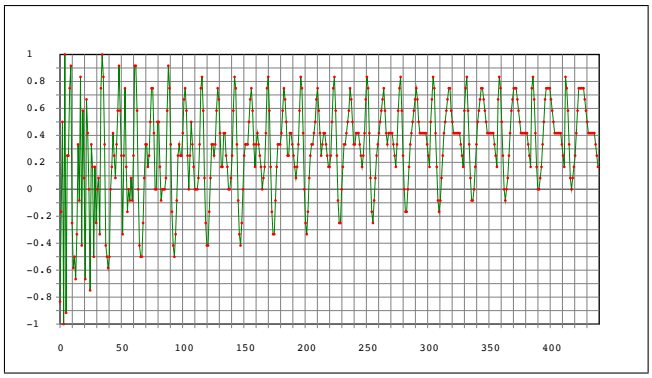

Figure 16. Recursively filtered signal used as a source for *Daripessus Yantillipicus*.

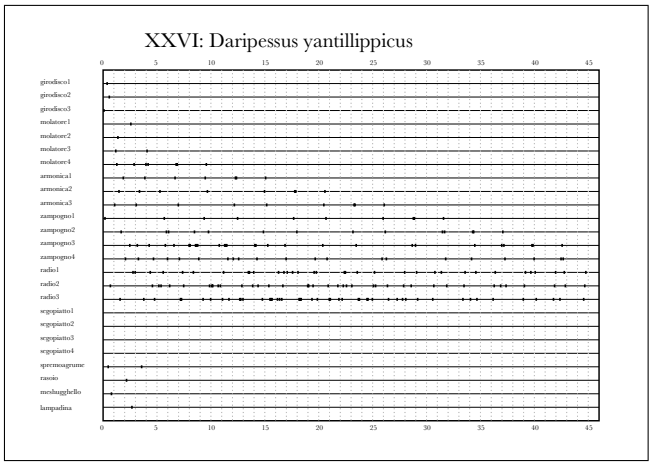

Figure 17. Visualisation score for electromechanical devices in *Daripessus Yantillipicus*.

were produced via NoteAbility Pro [4], an advanced music notation software that includes GUI and playback. NoteAbility Pro can import files written in the compact Guido textual description format [20] and has a much more flexible user frontend compared to the tools available for LilyPond. Thus, a SuperCollider module (GuidoGen in Figure 15) has been implemented, that generates files in the Guido format, so that they can be imported into NoteAbility Pro, played back, eventually re-edited, finally exported as graphic notation files.

4. APPLICATIONS TO OTHER TIME-DOMAIN BEHAVIOURS

Audio DSP techniques provide a vast repertoire of time-patterns that can be used to represent various behaviours on different time scales. In order to extend the previous discussion, this section presents some of the procedures discussed above in relation to the *Systema naturae* cycle for acoustic instruments and electromechanical devices (2013-17), composed by the author and Mauro Lanza [21, 22]. The cycle is made up of four pieces (*Regnum animale, Regnum vegetabile, Regnum lapideum, Fossilia*) and integrates various instrumental ensembles with real-time, computer-controlled, electromechanical devices, in the perspective of physical computing [23].

[4] `http://debussy.music.ubc.ca/NoteAbility/`

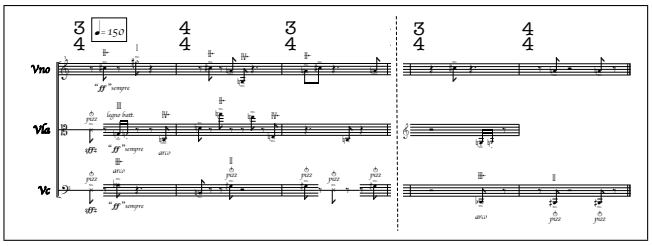

Figure 18. Beginning and ending measures from the instrumental score of *Daripessus Yantillipicus*.

In all the four setups of the cycle, the electromechanical devices are obtained from low-cost, scavenged/recycled objects and materials. In order to produce sounds, they are equipped either with small 12 volt DC motors, that provide mechanical energy to scrape, beat, pluck, or blow various objects and materials, or with 12 volt relais, that switch other circuits on/off (e.g. in order to scrape metal plates, they turn on/off modified electric knives operating at 230 volt). While relais work in a binary mode (on/off), DC motors can be fed with a variable amount of current modulated (on a 8 bit scale) by means of Arduino microcontrollers' Pulse Width Modulation (PWM) circuitry [24]. While operating in real-time, in *Systema naturae* these devices follow a pre-composed score that allows a strict synchronisation with human players, as the latter follow a click track. In *Systema naturae*, each of the four works of the cycle is in itself a collection of short pieces. Among these, various have been composed using SampComp techniques, applied both to melodic construction (see the example at the end of Section 2) and to device control.

In *Regnum animale*, the piece *Daripessus Yantillipicus* exploits the recursive noise filtering approach discussed above, to define the "melodic" sequence of activation of the devices. This means that values of audio samples are quantized so that they can be associated to the devices, as the latter are ranked in a list. The recursively filtered source signal used to activate deviced is shown in Figure 16 (DC offset is not an issue, as src will be mapped onto mld). The visualisation score is shown in Figure 17 [5]. In this case, activation for each device is on/off. The noisy attack in the src signal thus results in a dispersion of events, while filtering reduces the range of activated events. In the same piece, such an activation sequence is orchestrated for traditional instruments too. Figure 18 shows the first three and the last two measures of the instrumental parts (separated by a dashed line), so that the filtering from noise-based dispersion of events to a compressed range is apparent.

The same mapping approach (sample values onto ranked devices) is exploited in *Ismiosia Papanabuis* from *Regnum vegetabile*, an example interesting also from a general per-

[5] Score for devices are automatically generated from the composition data structures by exporting PostScript code, following the approach discussed before. From top to bottom, the names of the devices are: girodisco1, girodisco2, girodisco3, molatore1, molatore2, molatore3, molatore4, armonica1, armonica2, armonica3, zampogno1, zampogno2, zampogno3, zampogno4, radio1, radio2, radio3, segopiatto1, segopiatto2, segopiatto3, segopiatto4, spremoagrume, rasoio, meshugghello, lampadina.

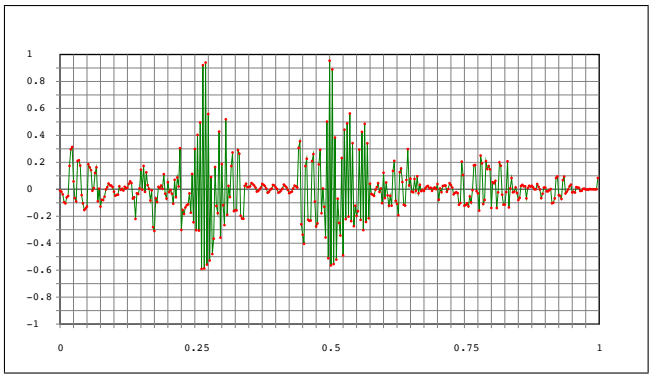

Figure 19. Audio sample (1 second) used as a source signal for *Ismiosia Papanabuis*.

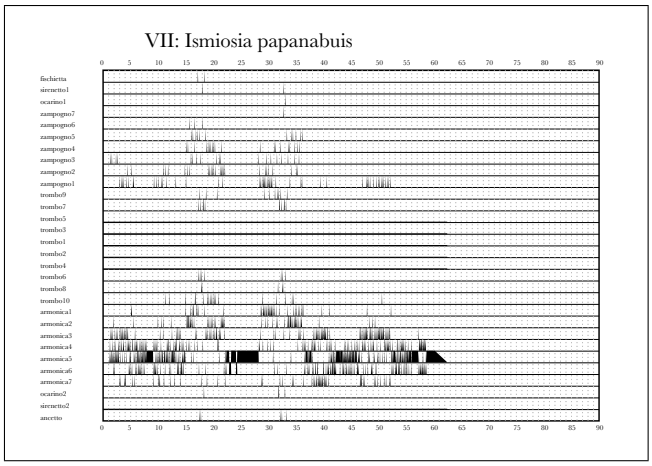

Figure 20. Visualisation score for electromechanical devices in *Ismiosia Papanabuis*.

spective. In this case, a short audio fragment is mapped directly onto events, like in *Daripessus*. The 1-second audio sample (Figure 19) is taken from the recording of an electroacoustic improvisation made by the author and Simone Pappalardo, and it has been chosen both as a formal model for composition because of its temporal behaviour and as a homage to Pappalardo, a frequent collaborator of both composers. The only operation performed by MapFunc is the quantization from values in the audio signal to indices representing the selected devices in a ranked list. Figure 20[6] shows the visualisation score for devices. Device activations describe a single melodic contour, following the imported waveform (compare Figure 19 and Figure 20). In the piece, the same principle is applied to acoustic instruments, so that the overall composition is properly a *hochetus* (if attacks, rather that durations, are taken into account), as Figure 21 shows.

As discussed, the electromechanical devices (in case of DC motors) expose as a control interface for their behaviour a single parameter, the 8-bit value addressable for PWM signals. In this sense, they fit the audio signal, as

[6] From top to bottom, devices are: fischietta, sirenetto1, ocarino1, zampogno7, zampogno6, zampogno5, zampogno4, zampogno3, zampogno2, zampogno1, trombo9, trombo7, trombo5, trombo3, trombo1, trombo2, trombo4, trombo6, trombo8, trombo10, armonica1, armonica2, armonica3, armonica4, armonica5, armonica6, armonica7, ocarino2, sirenetto2, ancetto.

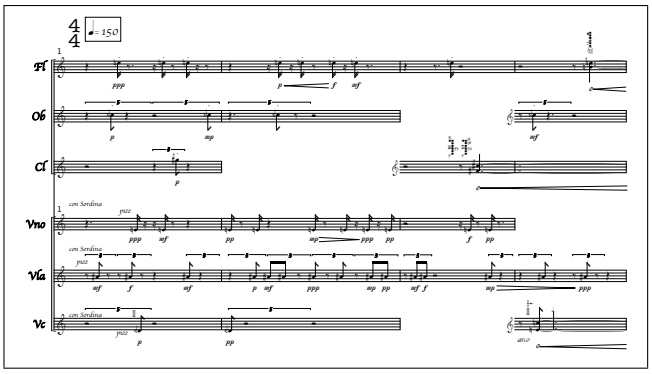

Figure 21. Beginning measures from the instrumental score of *Ismiosia Papanabuis*.

both the PWM and the audio signal are one-dimensional. While in previous examples, all devices were treated as operating in a binary mode, such a PWM control signal can be directly generated via SampComp techniques. The following two examples are from *Regnum vegetabile* too. The average filtering technique is used in the piece *Ariolactus usteginsiphillemena* with a different approach. In the piece, filtering is used to manage the delivery of electric current to group of devices, thus properly implementing a Karplus-Strong-like algorithm that simulates an attack/resonance acoustic model to generate 8-bit PWM signals. Figure 22 reproduces the visualisation score for machines. In this case, the averaging filter is used to directly control the values sent by Arduino's PWM circuitry to devices. Finally, the same approach is at play in the piece *Reocerantroma phenaudi*. PWM signals for devices are obtained by applying the previously introduced technique of the inverted sinusoid with impulse replacement, yielding to activation patterns made up of a continuous behaviour with short spikes (as can be clearly seen in Figure 23).

5. CONCLUSIONS

Audioparity can be seen as a musical aesthetics meta-category that groups together many historically and technically different approaches to composition, that share a common interest in sound as a starting point for music creation. Thus, SampComp can be seen as an audioparous approach, provided that sound as as a starting point is not considered neither in perceptual nor in acoustic terms, rather in relation to its technological implementation as a digital audio signal. This allows to profit from two mutually related aspects. On one side, the digital encoding of information allows a continuous transition from the sample level to the event one, due to the common numerical format. On the other one, by taking into account digital audio signals, a large corpus of knowledge is available to experimentation from the DSP theoretical domain. In short, the audio signal is taken into account as an abstract cognitive form to describe time patterns. Hence, the possible interest of SampComp techniques, which use can be further expanded and not necessarily limited to pitch contour generation: as an example, to manage the harmonic level or, as discussed in the last section, to generate time forms for generic sound

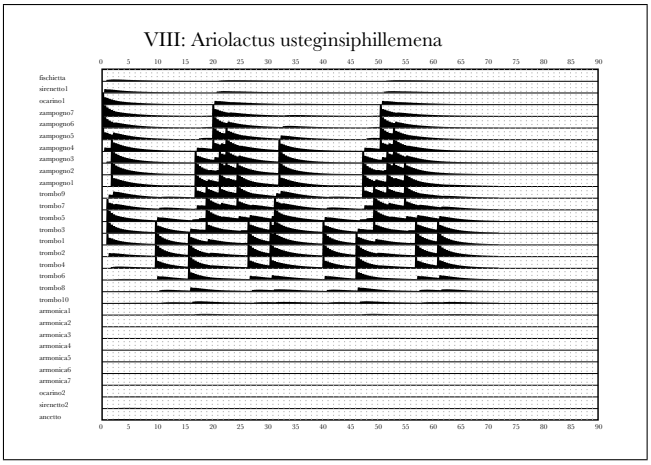

Figure 22. Visualisation score for electromechanical devices in *Ariolactus usteginsiphillemena*.

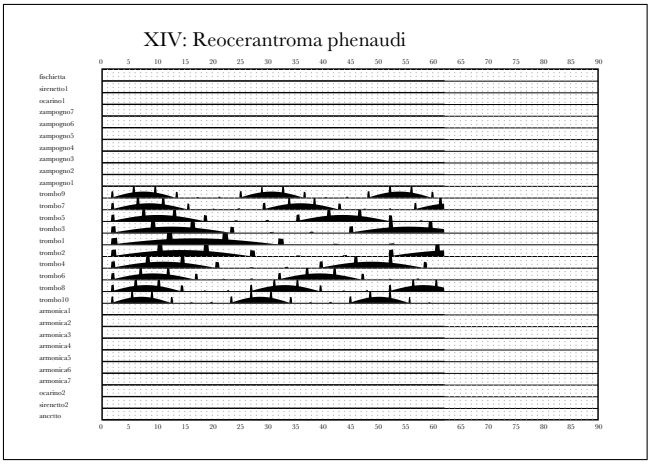

Figure 23. Visualisation score for electromechanical devices in *Reocerantroma phenaudi*.

producing behaviours.

6. REFERENCES

[1] N. Chomsky, *Syntactic Structures*. The Hague: Mouton and Co., 1957.

[2] M. A. Boden and E. A. Edmonds, "What is generative art?," *Digital Creativity*, vol. 20, no. 1-2, pp. 21–46, 2009.

[3] O. Messiaen, *The Techniques of my Musical Language*. Paris: Leduc, 1956.

[4] A. Orcalli, *Fenomenologia della musica sperimentale*. Potenza: Sonus, 1993.

[5] F. Paris, *Le temps de l'écoute. Gerard Grisey ou la beauté des ombres sonores*, ch. L'empreinte du cygne, pp. 51–65. Paris: L'Harmattan, 2004.

[6] K. Stockhausen, "...how time passes," *die Reihe*, vol. 3, pp. 10–43, 1957.

[7] C. Roads, *Microsound*. Cambridge, Mass. and London: The MIT Press, 2001.

[8] I. Xenakis and R. Brown, "Concerning time," *Perspectives of New Music*, vol. 27, no. 1, pp. 84–92, 1989.

[9] C. Roads, *The Computer Music Tutorial*. Cambridge, MA, USA: MIT Press, 1996.

[10] A. Strange, *Electronic Music: Systems, Techniques, and Controls*. Dubuque, Iowa: William C Brown Company, 2nd ed., 1983.

[11] M. Puckette, *The Theory and Technique of Electronic Music*. River Edge, NJ, USA: World Scientific Publishing Co., Inc., 2007.

[12] A. Bregman, *Auditory Scene Analysis. The Perceptual Organization of Sound*. Cambridge, Mass. and London: The MIT Press, 1990.

[13] P. Horowitz and W. Hill, *The Art of Electronics*. New York, NY, USA: Cambridge University Press, 3rd (2015) ed., 1989.

[14] D. Rocchesso, *Introduction to Sound Processing*. Firenze: Mondo Estremo, 2003.

[15] K. Karplus and A. Strong, "Digital synthesis of plucked-string and drum timbres," *Computer Music Journal*, vol. 7, no. 2, pp. 43–55, 1983.

[16] S. Wilson, D. Cottle, and N. Collins, eds., *The SuperCollider Book*. Cambridge, Mass.: The MIT Press, 2011.

[17] A. Valle, "Integrated Algorithmic Composition. Fluid Systems for including notation in music composition cycle," in *NIME 2008: Proceedings*, pp. 253–256, 2008.

[18] Adobe, *PostScript Language Reference*. Reading, Mass.: Addison-Wesley, 3rd ed., 1999.

[19] H.-W. Nienhuys and J. Nieuwenhuizen, "LilyPond, a system for music engraving," in *Proceeding of the XIV CIM 2003*, (Firenze), pp. 167–172, 2003.

[20] H. H. Hoos, K. A. Hamel, K. Renz, and J. Kilian, "The GUIDO Notation Format – A Novel Approach for Adequately Representing Score-Level Music," in *ICMC'98 Proceedings*, pp. 451–454, 1998.

[21] P. Roullier, ed., *Mauro Lanza et Andrea Valle: Systema naturae*. Champigny-sur-Marne: Ensemble 2e2m, 2016.

[22] A. Valle and M. Lanza, "Systema naturae: shared practices between physical computing and algorithmic composition," in *Proceedings of the 14th Sound and Music Computing Conference* (J. P. Tapio Lokki and V. Välimäki, eds.), (Espoo), pp. 391–398, Aalto University, Aalto University, 2017.

[23] D. O'Sullivan and T. Igoe, *Physical Computing. Sensing and Controlling the Physical World with Computers*. Boston, Mass.: Course Technology, 2004.

[24] M. Banzi, *Getting started with Arduino*. O'Reilly, 2009.